Remarks:

Reconsideration of the application is respectfully requested.

Claims 1 - 7 are presently pending in the application. Claim 6 has been amended.

In item 6 of the above-identified Office Action, claims 6 and 7 were rejected as being indefinite under 35 U.S.C. § 112, first paragraph. More specifically, it was alleged in the Office Action that:

> "[m]ultiple program counters each having the capability to selectively add or subtract an instruction length or to not add or subtract an instruction length to the program counter is not disclosed. Also, multiple program counters being used with the selective adding or subtracting of an instruction length to an offset was not disclosed. It appears applicant combined two or three different embodiments, one with multiple program counters and one with selectively adding/subtracting an instruction length to an offset value. However this was not disclosed in the specification."

Applicants respectfully disagree with the above statement in item 6 the Office Action. Rather, the instant application does describe an embodiment wherein a plurality of program counters is provided and the correct program counter is selected and provided to the proper computation unit. More particularly, page 4 of the instant application, line 1 - page 5, line 5, discloses:

> "According to the invention, it is possible in this case for example to provide a plurality of program counters and, dependent on the parameter, in each case activate one of the program counters for the computation of the relative addresses.
>
> In this case, it is particularly preferred according to the invention that the program counters are connected to a multiplexer, that is controlled by the parameter, and the output of the multiplexer is connected to the computation unit for the relative addresses. In this way, the selection of the correct program counter can take place very easily.
>
> Furthermore, the correct relative addressing can be ensured according to the invention by there being disposed between the program counter and the computation unit for the relative addresses an adding unit. One input of which is connected to the program counter and the other input of which is connected via a multiplexer, which is controlled by the parameter, to a memory for an instruction length or to the value 0, and the output of which is connected to the computation unit.
>
> Similarly, a correct relative addressing can be achieved according to the invention by there being disposed between the program counter and the computation unit for the relative addresses a subtracting unit. One input of the subtracting unit is connected to the program counter and the other input of which is connected via a multiplexer, which is controlled by the parameter, to a memory for the instruction length or to the value 0, and the output of which is connected to the computation unit." [emphasis added by Applicants]

As such, Applicants believe that the instant application has precisely disclosed, in a single embodiment, the use of multiple program counters, each having the capability to selectively add or subtract an instruction length or to not add or subtract an instruction length to the program counter.

Additionally, on page 5 of the instant application, line 18 - page 6, line 4, it is stated:

> "Similarly, it is preferably possible according to the invention, **dependent on the various operating states or assembler codes, to add or subtract the instruction length to or from the program counter reading** for the relative address computation or to leave the program counter reading unchanged/
>
> Similarly, it is possible according to the invention, **dependent on various operating states or assembler codes, to add, or subtract, an instruction length to or from the offset value,** which is usually used for the computation of relative addresses, or to leave the offset value unchanged in each case." [emphasis added by Applicants]

As such, the specification of the instant application clearly discloses, that Applicants' disclosed the **system** of page 4, line 1 - page 5, line 5, can, **both, add or subtract the instruction length to or from the program counter reading and add or subtract an instruction length to or from the offset value, in dependence on various operating states or assembler codes**. As such, Applicants' respectfully disagree with the statement in item 6 of the Office Action, and believe that the above-quoted portions of the instant application properly support and disclose the subject matter of Applicants' claims 6 and 7. It is accordingly believed that the claims 6 and 7 meet the requirements of 35 U.S.C. § 112, first paragraph.

In item 7 of the above-identified Office Action, claims 6 and 7 were rejected as being indefinite under 35 U.S.C. § 112,

second paragraph. More specifically, it was alleged in the Office Action that in claim 6, "leaving the program counter" is claimed and that, because of previously recited program counters, it is unclear which "program counter reading is unchanged". The Office Action additionally alleged that the above phrase is unclear, "since there is no teaching of multiple program counters selectively having an instruction length added/subtracted to them in the specification." As stated above in connection with item 6 of the Office Action, "multiple program counters selectively having an instruction length added/subtracted" is explicitly set forth on page 4, line 1 - page 5, line 5 of the instant application.

However, in order to make claims 6 and 7 even more clear, claim 6 has been amended to more particularly recite that the program counter is the one selected in the "selecting" step of claim 5. Support for these changes may be found on page 4, line 1 - page 5, line 5 of the specification of the instant application.

It is accordingly believed that the specification and the claims meet the requirements of 35 U.S.C. § 112, first and second paragraphs.

In item 9 of the Office Action, claims 1, 2 and 5 - 7 were rejected under 35 U.S.C. § 103(a) as allegedly being obvious over U. S. Patent No. 4,821,183 (mistakenly listed in the Office Action as "7,821,183") to Hauris ("**HAURIS**") in view of U. S. Patent No. 5,781,750 to Blomgren et al ("**BLOMGREN**").

In item 19 of the Office Action, claims 3 and 4 were rejected under 35 U.S.C. § 103(a) as allegedly being obvious over U. S. Patent No. 5,854,913 to Goetz et al("**GOETZ**") in view of U. S. Patent No. 5,088,030 to Yoshida ("**YOSHIDA**"), U. S. Patent No. 4,926,323 to Baror et al("**BAROR**"), The PowerPC Architecture, 1994 ("**POWERPC**") and K. Short, Embedded Microprocessor Systems Design, 1998 ("**SHORT**").

Applicants respectfully traverse the above rejections.

I. **Applicants' independent claims 1 and 5 require, among other limitations, various assembler codes, and "a computation of relative addresses" for each of the respective assembler codes.**

Applicants' claim 1 recites, a microprocessor for processing **various assembler codes**, comprising:

> **a parameter designating a respective assembler code** and, **depending on how the parameter is set, a different relative addressing takes place**; and
>
> a plurality of program counters and, **dependent on the parameter, for each of said respective assembler codes**

> **one of said program counters is active in a computation of relative addresses."** [emphasis added by Applicants]

Applicants' independent claim 5, similarly recites, a method of **relative addressing** in a microprocessor, which comprises the steps of:

> **"determining relative addresses** <u>**in dependence on one of an operating state and a parameter for a respective assembler code**</u>;
>
> providing a plurality of program counters <u>**for various operating states and assembler codes**</u>; and
>
> **selecting one of the program counters for use in determining the relative addresses** <u>**in dependence on one of the operating state and the respective assembler code**</u>." [emphasis added by Applicants]

As such, Applicants' claims require, among other limitations, **relative addressing** in a microprocessor. Applicants' claim 1 particularly requires, among other limitations, that the **relative addressing** occur based on a **parameter** designating <u>**a respective assembler code**</u> from <u>**various assembler codes**</u>. Similarly, Applicants' independent claim 5 requires, among other things, **relative addressing** for utilizing <u>**various operating states and assembler codes**</u>.

Applicants' claimed **relative addressing** in a system allows the use in a single system of **various assembler codes** (i.e., claims 1 - 2 and 5 - 7) and/or **various operating states** (i.e.,

claims 5 - 7) as stated, on page 14 of the instant application, lines 9 - 13:

> "According to the invention, it is consequently possible for the first time to realize a processor which allows different assembler programming languages with different computation specifications for relative destinations in relation to the instruction counter within a CPU."

It is the **relative addressing** process of the current invention that makes it possible for a processor of the invention of claims 1 and 5 to run different assembler codes and/ or operating states, as specified on page 5 of the instant application, lines 7 - 11:

> "To achieve the object according to the invention, the present invention further teaches a method of relative addressing in a microprocessor in which, dependent on an operating state or parameter for the respective assembler code, relative addresses are differently determined."

The **relative addressing** that permits the use of **various assembler codes** in a single microprocessor, as claimed in Applicants' independent claims 1 and 5 is additionally described on page 11 of the instant application, line 18 - page 12, line 6, which states:

> **"Since the assembler codes in modern microprocessor systems can be stored at various locations in the main memory** before they are processed by the microprocessor, **it is required to give addressings in a relative form**, that is to say with respect to the respective configuration of the assembler code in the main memory. **Relative addressing,** in which a specific offset value is

> additionally computed for all the instructions that are **pointing to a different address in the assembler code**, serves for this purpose. The offset value usually corresponds to the distance of the assembler code in the main memory, the distance at which the program has been stored away from the operating system. By use of the offset, the relative branch addresses present in the assembler code can then be assigned to the actual physical memory locations of the respective program line." [emphasis added by Applicants]

As such, in the invention of claims 1 and 5, as supported by the instant specification, the claimed **relative addressing** permits the processor of the invention to **choose between** (i.e., **point** between) **various different assembler codes and/or operating states** simultaneously contained in memory, with which to perform an operation. The desired **assembler codes** and/or **operating state** is chosen from among the plurality, based on Applicants' claimed relative addressing process of claims 1 and 5.

II. Neither HAURIS, nor BLOMGREN, disclose, among other limitations of Applicants' claims, "a computation of relative addresses" based on a particular assembler code/operating state of various assembler codes/operating states.

On page 4 of the Office Action, it is alleged that **HAURIS** discloses a microprocessor for processing various assembler codes. However, on page 5 of the Office Action, it is admitted in item 11, that:

> "[w]hile Hauris does teach a parameter selecting between program counters to access subroutines, he

> **fails to teach wherein the subroutines <u>are multiple assembler codes</u>.**" [emphasis added by Applicants]

As such, it is admitted in the Office Action that **HAURIS** does not teach relative addressing of multiple assembler codes.

The Office Action goes on, in item 12, to allege:

> "Blomgren teaches:
> - Multiple assembler codes being processed (x86 instruction set and PowerPC instruction set): Blomgren teaches that when a complex x86 instruction is reached, a new instruction pointer is loaded to indicate a software subroutine made up of PowerPC instructions."

Further, in item 13 of the Office Action it was alleged that:

> "It would have been obvious to one of ordinary skill in the art to combine the invention if Hauris with the multiple assembly codes of Blomgren because as Blomgren states, 'Since there is so much installed x86 code, it is greately desired to run x86 programs on newer RISC CPU's, without the performance degradation of a software emulator,' column 3, lines 35 - 38. **Blomgren does this by using a RISC and CISC decoder, however, for the more complex x86 instructions, he uses a different instruction pointer to access a subroutine of RISC instruction in a ROM.** Hauris already teaches accessing a ROM for a subroutine using multiple program counters. It would have been obvious to have the processor decode and execute both RISC and CISC instructions as taught in Blomgren by having decoders for both instructions sets, and using the ROM subroutine accessing method of Hauris to process the more complex CISC instructions. This would cause said parameter of Hauris to indicate to switch assembly codes because the processor would be in a CISC mode until reaching a complex CISC instruction, at which point a RISC subroutine would be accessed via another program counter." [emphasis added by Applicants]

Applicants' respectfully disagree with the allegations made in the Office Action, and more particularly: a) disagrees that **BLOMGREN** shows "**a different instruction pointer to access " different assembler codes**; and b) further disagrees that combining **BLOMGREN** with **HAURIS**, would teach or suggest Applicants' invention of claims 1 and 5.

The **BLOMGREN** reference discloses a dual-instruction set architecture CPU with hidden software emulation mode. In **BLOMGREN**, **does not** use **relative addressing to choose between** (i.e., **point** between) **various different assembler codes and/or operating states** simultaneously contained in memory, **with** which to perform an operation, as required by Applicants' claims 1 and 5. Rather, as taught in , col. 5 of **BLOMGREN**, lines, 58 - 61:

> "A software routine will be executed that **breaks the complex CISC instruction down into several smaller RISC instructions**, such as Loads and Stores." [emphasis added by Applicants]

As such, **BLOMGREN** does not use **relative addressing** to process instructions in one of a plurality of stored **various assembler codes**. Instead, **BLOMGREN** breaks down operations from different command sets **into a single command set (i.e. RISC)**, and uses that **single command set** to process instructions, regardless of the actual command and its particular original command set. This **single command set** is used to process

everything in **BLOMGREN** with the disclosed hardware. If there is no command in the single command set (i.e., RISC) of hardware to address a particular command, as disclosed in col. 5 of **BLOMGREN**, lines 55 - 58,

> "Emulation mode executes routines that emulate the behavior of the complex instructions that are not supported directly by the hardware. . . .A software routine will be executed **that breaks the complex CISC instruction down into several smaller RISC instructions**, such as Loads and Stores." [emphasis added by Applicants]

Thus, **BLOMGREN** fails to teach or suggest, the computation of **relative addressing** based on a particular assembler code/operating state of various assembler codes/operating states. Rather, **BLOMGREN** only uses the RISC command set to process instructions. This is further supported in the Abstract of **BLOMGREN**, which states:

> "Software routines to emulate the more complex instructions of the CISC architecture **using the RISC instructions** are also loaded into the emulation memory." [emphasis added by Applicants]

As such, **BLOMGREN**, alone or in combination with **HAURIS**, fails to teach or suggest, among other limitations of Applicants' invention of claims 1 -2 and 5 - 7, Applicants' particular **relative addressing to access respective ones of different assembler codes/operating states in the memory.**

As such, claims 1 - 2 and 5 - 7 of the instant application are believed to be allowable over the **BLOMGREN** and **HAURIS** references.

III. **Applicants' independent claims 3 and 4 require, among other limitations, processing various assembler codes, including a multiplexer "receiving a parameter designating a respective assembler code and, depending on how the parameter is set, a different relative addressing takes place".**

Applicants' independent claims 3 and 4 recite, among other limitations, a microprocessor for processing various assembler codes, comprising:

> "a multiplexer having a first input, a second input for receiving a 0 value, and a third input receiving a parameter designating a respective assembler code and, depending on how the parameter is set, a different relative addressing takes place;
>
> a program counter;
>
> a computation unit for computing relative addresses;
>
> . . .
>
> a memory for storing an instruction length and having an output connected to said first input of said multiplexer." [emphasis added by Applicants]

As such, as described above in connection with Applicants' claims 1 and 5, in Applicants' claims 3 and 4, there is claimed a microprocessor for processing various assembler codes including a multiplexer used to implement a particularly claimed relative addressing process, based on which of the

various assembler codes is designated.. The discussion of such relative addressing process, and of the processing of the various assembler codes based on the addressing of such process is discussed above in Section I, incorporated herein.

Applicants' independent claim 3, additionally recites, among other limitations:

> "an adding unit connected between said program counter and said computation unit, said adding unit having a first input connected to said program counter, a second input connected to said multiplexer, and an output connected to said computation unit;"

Applicants' independent claim 4, additionally recites, among other limitations:

> "an subtracting unit connected between said program counter and said computation unit for the relative addresses, said subtracting unit having a first input connected to said program counter, a second input connected to said multiplexer, and an output connected to said computation unit;"

Thus, output to the computation unit of claims 3 and 4 is generated based on input from the program counter and the multiplexer.

IV. The cited references fail to teach or suggest, among other limitations of Applicants' claims 3 and 4, processing various assembler codes, including a multiplexer "receiving a parameter designating a respective assembler code and, depending on how the parameter is set, a different relative addressing takes place".

In the Office Action, it is alleged that claims 3 and 4 were obvious over **GOETZ** in view **YOSHIDA, BARON, POWERPC** and **SHORT**. Applicants respectfully disagree. More particularly, in item 21 of the Office Action, it was alleged that:

> **"While Goetz does teach multiple instruction sets being implemented and indicated by a parameter,** Goetz is silent on how the different offsets for branch instructions are handled." [emphasis added by Applicants]

However, Applicants respectfully disagree with the allegation that **GOETZ** teaches multiple instruction sets being implemented and indicated by a parameter, in a manner applicable to Applicants' claims 3 and 4. Rather, instead of using Applicants' claimed relative addressing process, including the claimed multiplexer, program counter and computation unit, **GOETZ** offers a much different solution. Contary to Applicants' invention, **GOETZ** provides <u>two separate</u> **command set architectures running in parallel, and uses memory management hardware to switch between them. See GOETZ**, col. 5, lines 1 - 7. Further, as stated in col. 5 of **GOETZ**, lines 21 -43:

> "According to the invention there is provided a microprocessor which runs under a single multitasking operating system **and supports first and second architectures <u>having separate and distinct instruction sets and memory management schemes</u>**. The microprocessor comprises instruction set management means that decodes instructions in a first instruction set of the first architecture and decodes instructions of a second instruction set of the second architecture. The

> instruction set management means maps decoded instructions in the first instruction set to one or more instructions in the second instruction set. **The microprocessor further includes memory management means that performs address translation from virtual to real addresses for said first and second architectures.** Control means detects an architectural context of a program being read from memory as being either code for said first architecture or code for said second architecture and, depending on the detected architectural context, **controls the instruction set management means and said memory management means to dynamically switch between address translation for the first or second architectures** and executing one or more mapped decoded instructions or directly decoded instructions of the second architecture." [emphasis added by Applicants]

As such, **GOETZ** teaches away from Applicants' claimed invention of claims 3 and 4. Were the teachings of **GOETZ** to be modified in combination with the other references, as suggested in the Office Action, absent hindsight reconstruction, a person of skill in the art would still not obtain Applicants' invention of claims 3 and 4. More particularly, using the teachings of **GOETZ, each of the two architectures of GOETZ** would contain its own program counter (i.e., two program counters each having a different relative addressing), used to jump to/activate the next command address in **that particular architecture's** command set, when that particular architecture is enabled (i.e., switched to). Thus, **GOETZ neither teaches, nor suggests, Applicants' claimed particularly claimed "program counter"** or its particularly claimed use in connection with an adding/subtracting unit for calculating between (i.e., pointing between) the addresses for the various

different assembler codes in a memory. And as a result, **GOETZ teaches away from Applicants' particularly claimed program counter, and its use in the invention of claims 3 and 4 (i.e., as an input to the adding/subtracting unit).**

As such, **GOETZ** does not teach or suggest, alone or in combination with the other cited references, Applicants' invention of claims 3 and 4.

None of the other references cited in the Office Action, make up for the deficiencies or the teaching away from Applicants' claimed invention of the **GOETZ** reference.

V. Conclusion.

It is accordingly believed that none of the references, whether taken alone or in any combination, teach or suggest the features of claims 1, 3, 4 and 5. Claims 1, 3, 4 and 5 are, therefore, believed to be patentable over the art. The dependent claims are believed to be patentable as well because they all are ultimately dependent on claims 1 or 5. As it is believed that the claims were patentable over the cited art in their original form, the claims have not been amended to overcome the references.

In view of the foregoing, reconsideration and allowance of claims 1 - 7 are solicited.

In the event the Examiner should still find any of the claims to be unpatentable, counsel would appreciate receiving a telephone call so that, if possible, patentable language can be worked out.

Additionally, please consider the present as a petition for a one month extension of time, and please provide a one month extension of time, to and including, May 13, 2005 to respond to the present Office Action.

The extension fee for response within a period of one month pursuant to Section 1.136(a) in the amount of $120.00 in accordance with Section 1.17 is enclosed herewith.

Please provide any additional extensions of time that may be necessary and charge any other fees that might be due with respect to Sections 1.16 and 1.17 to the Deposit Account of Lerner and Greenberg, P.A., No. 12-1099.

Applic. No. 09/928,011
Response Dated May 13, 2005
Responsive to Office Action of January 13, 2005

Respectfully submitted,

Kerry P. Sisselman
Reg. No. 37,237

For Applicants

KPS:cgm

May 13, 2005

Lerner and Greenberg, P.A.
Post Office Box 2480
Hollywood, FL 33022-2480
Tel: (954) 925-1100
Fax: (954) 925-1101